Sudoku solver

# Objectives

## Implement a Sudoku solver.

[Sudoku](https://en.wikipedia.org/wiki/Sudoku) is a logic-based, combinatorial number-placement puzzle. The objective is to fill a 9×9 grid with digits so that each column, each row, and each of the nine 3×3 subgrids that compose the grid (also called "boxes", "blocks", or "regions") contain all of the digits from 1 to 9. The puzzle setter provides a partially completed grid, which for a well-posed puzzle has a single solution.

Sudoku solver accepts a Sudoku board, tries to fill in missing numbers according to the rules, returns solving status - success/failure, and result - completed Sudoku board.

## Test a Sudoku solver

Sudoku solver should be tested on at least two examples provided (see [assumptions](#_Assumptions)).

To evaluate algorithm effectiveness use a [Sudoku designed to work against brute force algorithms](https://en.wikipedia.org/wiki/Sudoku_solving_algorithms#/media/File:Sudoku_puzzle_hard_for_brute_force.svg).

# Requirements

* Use Java programming language for implementation.
* Do not use existing solutions (from Internet).
* Provide solution with its source code.

# Assumptions

## Algorithm

* There is no strict performance requirement thereby:
  + Backtracking based solution algorithm can be used as simple, fast to implement and easy to debug, which makes it a good MVP solution considering very limited input data size (9x9 grid).
  + No parallelization will be done, as it increases implementation complexity and brings less performance impact than some alternative algorithms.

Advantages of this method are:

* + A solution is guaranteed (as long as the puzzle is valid).
  + Solving time is mostly unrelated to degree of difficulty.
  + The algorithm (and therefore the program code) is simpler than other algorithms, especially compared to strong algorithms that ensure a solution to the most difficult puzzles.

## Abstractions

* Dimensions of the board will not change, but Sudoku board abstraction should be easily modifiable for other sizes.
* Implementation should allow considerably fast replacement of the algorithm by providing a clear interface and abstractions.

## User interface and input format

* MVP can be implemented as a console application
* As an input use a text file with 9 rows of 9 numbers (0-9), zero indicates an empty cell. Examples:

|  |  |
| --- | --- |
| 002000041  000082070  000040009  200079300  010000080  006810004  100090000  060430000  850000400 | 000000000  000003085  001020000  000507000  004000100  090000000  500000073  002010000  000040009 |

* Sudoku solver should not fail on degenerate (all zeros) or unsolvable boards.
* For boards allowing more than one solution solver may return the first found solution.